

Lossy Source Compression of Non-Uniform Binary Sources Using GQ-LDGM Codes

Lorenzo Cappellari

Dept. of Information Engineering, University of Padova, Italy

Email: lorenzo.cappellari@dei.unipd.it

Abstract—In this paper, we study the use of $GF(q)$ -quantized LDGM codes for binary source coding. By employing quantization, it is possible to obtain binary codewords with a non-uniform distribution. The obtained statistics is hence suitable for optimal, direct quantization of non-uniform Bernoulli sources. We employ a message-passing algorithm combined with a decimation procedure in order to perform compression. The experimental results based on $GF(q)$ -LDGM codes with regular degree distributions yield performances quite close to the theoretical rate-distortion bounds.

I. INTRODUCTION

In the proof of achievability of the channel capacity, as well as in the proof of achievability of the rate-distortion function, random ensembles of codes are tested and shown to be asymptotically optimal [1]. But, in practice, random codes are useless because without any structure no channel decoding/source encoding algorithms exist that run in reasonable time. In many cases of interest (e.g. binary symmetric channels, binary source coding with Hamming-distance distortion measure, ...) optimality is also achieved by linear codes. However, conducting a search among their codewords is still a tough task unless some other condition holds.

Luckily, *sparse* structures, e.g. linear codes with parity-check or generator matrices with a number of non-zero elements that is roughly proportional to the length of their codewords, have been shown to asymptotically perform very close to the theoretical bounds. Notably, for these structures, there often exist practical algorithms that essentially permit to reach their asymptotic performances.

For example, in the case of channel coding for the binary symmetric channel, the *low density parity-check* (LDPC) codes are shown to almost achieve the channel capacity [2], at least under maximum-likelihood decoding. In addition, they can be decoded in linear time with respect to the codeword length by using the standard message-passing algorithm [3] with negligible performance losses [4].

On the other side, it has been proved that sparse codes are optimal too for lossy source coding. For example, *low density generator matrix* codes (LDGM) almost achieve the rate-distortion function of the binary-symmetric memoryless source with Hamming-distance distortion measure [5]; also, by appropriately combining LDPC and LDGM codes it is possible to solve problems of coding with side information [6]. But, if the standard message-passing algorithm [3] is employed in this dual setting, where there actually exist *multiple* near-

optimal encodings, it usually fails to converge or converges to meaningless marginals.

One of the first works showing that iterative algorithms can actually work as well for quantization appeared in [7], where the problem of *binary erasure quantization* over LDGM codes was tackled. In order to solve the convergence problem, a simple modification of the message-passing algorithm was proposed in [8] that led to satisfying results. Other works investigate adaptations of algorithms from the field of statistical physics in order to do data compression as well, e.g. by using the *survey propagation* algorithm [9]. Furthermore, practical iterative algorithms that rely on *non-linear* codes have been proposed for achieving the rate-distortion bound [10].

One interesting possible workaround to achieve convergence in binary source coding with LDGM codes, as suggested in [5], is to (i) run the traditional message-passing algorithm over an *augmented* code-space where variables can also take a *free* state, (ii) set the more certain variables to their preferred (non-free) states, and (iii) iterate this procedure over the *decimated* code obtained by eliminating the variables that have been set in step (ii), until all variables have been set. Indeed, this approach can be seen as extending both the standard message-passing rules and the survey propagation ones [11]; in addition, it can be simplified as shown in [12].

In this paper we aim at extending the algorithms described in [5] to the case in which the binary source to be compressed is not uniform. In this case, in fact, a linear binary code whose codewords have a uniform statistics is no longer optimal [1], as happens in the dual case of channel coding for non-symmetric channels. In the latter case optimality can still be achieved relying on non-binary LDPC codes and their variants, such as for example the $GF(q)$ -quantized *coset LDPC* (GQC-LDPC) codes [13]. Among others, GQC-LDPC codes have interesting applications in the framework of channel coding with side information at the encoder, where they are applied in conjunction with good source codes in a superposed fashion [14]. In particular, in this paper, we define $GF(q)$ -quantized LDGM (GQ-LDGM) codes as duals of GQC-LDPC codes, and design practical algorithms for quantization over their codewords. GQ-LDGM codes can be eventually employed in conjunction with good channel codes for source coding with side information at the decoder, using a superposition scheme dual to the one in [14].

As the question “can linear codes with a suitable mapping function achieve the rate-distortion bound?” was positively

answered in [15], and similar results have been proved for LDGM codes [16], our focus on the practical side of the problem, namely on designing practical encoding algorithms, is fully justified. Other than this, constructions based on *quantized* LDGM codes (or on *multilevel quantization*) were used as well to tackle scalable coding [17] and multiple description coding [18] problems. This paper is also connected with [10], where codes for non-uniform sources too are designed and tested.

The rest of this paper is organized as follows. In Section II we review the definition of $GF(q)$ -LDGM codes and extend the space-augmentation procedure used in [5] to these codes. In Section III we introduce GQ-LDGM codes and describe the message-passing and decimation rules used for performing quantization. In Section IV we present some experimental results obtained with regular degree GQ-LDGM codes. Section V summarizes our conclusions.

II. $GF(q)$ -LDGM CODES AND THEIR AUGMENTATION

An (n, m) LDGM code over $GF(q) = \{l_0 = 0, l_1, \dots, l_{q-1}\}$ is easily specified by means of a *factor graph* [3] made by n checks a and n constrained variables x_a , $a \in \{0, 1, \dots, n-1\}$, plus m free variables z_i , $i \in \{0, 1, \dots, m-1\}$. Each check a is connected to x_a on one side and to $\mathbf{z}_{N(a)}$ on the other¹, where $N(a) \subseteq \{0, 1, \dots, m-1\}$; each such connection is labelled with a *weight* $g_{ia} \in GF(q) \setminus \{l_0\}$. Each variable z_i is then connected to the checks in $N(i) \subseteq \{0, 1, \dots, n-1\}$. Variables take values on $GF(q)$, checks are satisfied iff $x_a = \sum_{i \in N(a)} g_{ia} z_i$, and the $(n+m)$ -tuples (\mathbf{x}, \mathbf{z}) such that all checks are satisfied form the codewords of the LDGM code.

The *augmented LDGM* code is made of all the $(n+m)$ -tuples of elements of $GF(q)^* \triangleq GF(q) \cup \{*\}$ such that, with reference to the factor-graph of the original code (see Fig. 1),

- 1) either checks connect to $x_a = *$ on one side and *at least* one element in $\mathbf{z}_{N(a)}$ equals $*$ on the other, or they are satisfied (i.e. their neighborhood is made of all elements of $GF(q)$ that satisfy the condition given above);
- 2) free variables different from $*$ connect to *at least* two exactly satisfied checks (we assume that $|N(i)| \geq 2$).

Those codewords are called *generalized* codewords.

Due to the constraint in 2, in order to check if an $(n+m)$ -tuple is a generalized codeword, not only additional *compatibility* functions must be imposed on the free variables, but also the corresponding nodes should be equipped with some knowledge about the state of the checks to which they connect. Similarly to what done in [5], this can be easily achieved by replacing the nodes corresponding to the free variables with some variables $P_i \in \mathcal{P}_i \triangleq GF(q) \times \{1, 2, \dots, 2^{|N(i)|} - 1\} \cup \{*\}$, where $\{1, 2, \dots, 2^{|N(i)|} - 1\}$ is assumed in one-to-one correspondence with the non-empty sets in $\mathcal{P}(N(i))$, the power set of $N(i)$. Once we define the two *projections*

¹ $\mathbf{z}_{N(a)}$ stands for $\{z_i : i \in N(a)\}$.

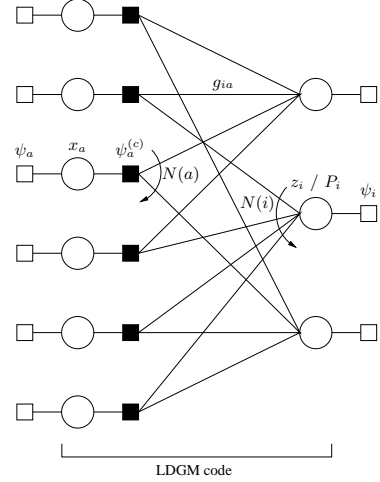


Fig. 1. Factor graph of an LDGM code completed with the compatibility functions for assigning suitable probabilities to its generalized codewords.

$\pi : \mathcal{P}_i \rightarrow GF(q)^*$ and $\varphi : \mathcal{P}_i \rightarrow \mathcal{P}(N(i))$ such that

$$\begin{aligned} \pi(P_i) &= \begin{cases} *, & \text{if } P_i = * \\ l_k, & \text{if } P_i = (l_k, \cdot) \end{cases} \\ \varphi(P_i) &= \begin{cases} \emptyset, & \text{if } P_i = * \\ A \subseteq N(i), & \text{if } P_i = (\cdot, \text{index of subset } A) \end{cases} \end{aligned}$$

and the compatibility functions at checks and nodes² (3) and

$$\psi'_i(P_i) = \chi[P_i = * \vee |\varphi(P_i)| \geq 2], \quad (2)$$

the generalized codewords are all the $(n+m)$ -tuples $(\mathbf{x}, \pi(\mathbf{P}))$ such that all x_a and P_i satisfy the new compatibility functions. In practice, P_i brings both the value of the free variable, and, in case of the value being different from $*$, the indices of the checks that are enforcing this value (note that at least one such check always exists).

III. BINARY SOURCE COMPRESSION WITH $GF(q)$ -QUANTIZED LDGM CODES

At this point, when one wants to use the original LDGM code in a quantized fashion for quantizing a random realization of a binary codeword, a map $Q : GF(q)^* \rightarrow GF(2)^*$ is introduced such that $Q(x_a) = *$ iff $x_a = *$. This map eventually converts a $GF(q)$ codeword corresponding (through standard LDGM encoding) to a compressed representation in $GF(q)$ into the $GF(2)$ codeword used as source reconstruction.

Then, by adding the compatibility functions for the constrained variables $\psi_a^{(Q)}(\bar{x}_a; s_a, \beta, w_s) = \psi_a^{(Q)}(Q(\bar{x}_a); s_a, \beta, w_s)$, where

$$\psi_a^{(Q)}(x_a^{(Q)}; s_a, \beta, w_s) = \begin{cases} e^\beta, & \text{if } x_a^{(Q)} = s_a \\ w_s, & \text{if } x_a^{(Q)} = * \\ e^{-\beta}, & \text{if } x_a^{(Q)} = s_a + 1 \end{cases}, \quad (3)$$

and by using $\psi_i(P_i; w_i) = w_i^{\chi[P_i = *]} \psi'_i(P_i)$ as compatibility functions for the free variables, a probability distribution over

² $\chi[\cdot]$ is one iff the condition indicated is satisfied and zero otherwise; it is also assumed that $l_k + * = * + l_k = *$ and $* + * = *$.

$$\begin{aligned} \psi_a^{(c)}(\bar{x}_a, \mathbf{P}_{N(a)}) &= \chi \left[\left(\sum_{i \in N(a)} \pi(P_i) = * \wedge \bar{x}_a = * \wedge \bigwedge_{i \in N(a)} x_a \notin \varphi(P_i) \right) \vee \right. \\ &\quad \left. \vee \left(\sum_{i \in N(a)} \pi(P_i) \neq * \wedge \bar{x}_a = \sum_{i \in N(a)} g_{ia} \pi(P_i) \wedge \bigwedge_{i \in N(a)} x_a \in \varphi(P_i) \right) \right] \end{aligned} \quad (3)$$

all $(n+m)$ -tuples (\mathbf{x}, \mathbf{P}) is enforced. This distribution, when marginalized over all the (\mathbf{x}, \mathbf{P}) tuples corresponding to the same generalized codeword (\mathbf{x}, \mathbf{z}) , leads to a probability

$$P[(\mathbf{x}, \mathbf{z}); s_a, \beta, w_s, w_i] \propto w_i^{N^*(\mathbf{z})} \cdot w_s^{N^*(\mathbf{x})} \cdot e^{-2\beta d_H(Q(\mathbf{x}), \mathbf{s})}. \quad (4)$$

The factor graph describing this distribution is shown in Fig. 1. In the formulas, \mathbf{s} denotes the n -dimensional source realization to be quantized, $\beta \geq 0$ determines how much more probability should be given to the values (taken by constrained variables) that after quantization match the source with respect to those that will not match it, and $w_s \geq 0$ and $w_i \geq 0$ determine how much each value $*$ (occurring in constrained or free variables, respectively) modifies the overall generalized codeword probability. In particular, $w_s > 1$ (or $w_i > 1$) will make the $*$ value to be preferred with respect to symbols in $GF(q)$. Also, $N^*(\cdot)$ denotes the number of $*$ values in a given tuple of $GF(q)^*$, and $d_H(\cdot, \cdot)$ denotes the Hamming distance between tuples of $GF(2)^*$, assuming that $*$ has distance 1/2 from any other symbol.

Despite the exponentially large cardinality of \mathcal{P}_i , that in practice seems to prevent the utilization of message-passing algorithms, it can be shown that the messages over each edge can be described by only $q+2$ probabilities, at least in the case where q is a *prime* number. In particular, in the check-to-variable direction these probabilities are μ_{ai}^* , μ_{ai}^\neq , and $\mu_{ai}^{(l_k)}$, for $k = 0, 1, \dots, q-1$, because the messages satisfy

$$\mu_{ai}(P_i = *) \equiv \mu_{ai}^* \quad (5)$$

$$\mu_{ai}(P_i : P_i \neq * \wedge x_a \notin \varphi(P_i)) \equiv \mu_{ai}^\neq \quad (6)$$

$$\mu_{ai}(P_i : \pi(P_i) = l_k \wedge x_a \in \varphi(P_i)) \equiv \mu_{ai}^{(l_k)}. \quad (7)$$

In the variable-to-check direction, it is similarly sufficient to specify the probabilities μ_{ia}^* , μ_{ia}^\neq , and $\mu_{ia}^{(l_k)}$, for $k = 0, 1, \dots, q-1$, which are defined as

$$\mu_{ia}(P_i = *) \triangleq \mu_{ia}^* \quad (8)$$

$$\sum_{P_i : P_i \neq * \wedge x_a \notin \varphi(P_i)} \mu_{ia}(P_i) \triangleq \mu_{ia}^\neq \quad (9)$$

$$\sum_{P_i : g_{ia} \pi(P_i) = l_k \wedge x_a \in \varphi(P_i)} \mu_{ia}(P_i) \triangleq \mu_{ia}^{(l_k)}. \quad (10)$$

With these definitions, the messages from check a to

variable i can be computed by

$$\mu_{ai}^* = w_s \prod_{j \in N(a) \setminus \{i\}} [\mu_{ja}^* + \mu_{ja}^\neq] \quad (11)$$

$$\mu_{ai}^\neq = w_s \left[\prod_{j \in N(a) \setminus \{i\}} [\mu_{ja}^* + \mu_{ja}^\neq] - \prod_{j \in N(a) \setminus \{i\}} \mu_{ja}^\neq \right]$$

$$\mu_{ai}^{(l_k)} = \sum_{r=0}^{q-1} \psi_a(l_r) \lambda_{ai}^{(l_r - g_{ia} l_k)}, \quad (12)$$

where³

$$[\lambda_{ai}^{(l_k)}] = \text{IDFT}_q \left[\prod_{j \in N(a) \setminus \{i\}} \text{DFT}_q[\mu_{ja}^{(l_k)}] \right]. \quad (13)$$

The messages from variable i to check a can be instead computed by

$$\mu_{ia}^* = w_i \prod_{c \in N(i) \setminus \{a\}} \mu_{ci}^* \quad (14)$$

$$\begin{aligned} \mu_{ia}^\neq &= \sum_{k=0}^{q-1} \prod_{c \in N(i) \setminus \{a\}} [\mu_{ci}^\neq + \mu_{ci}^{(l_k)}] - q \prod_{c \in N(i) \setminus \{a\}} \mu_{ci}^\neq - \\ &\quad - \sum_{b \in N(i) \setminus \{a\}} \left[\sum_{k=0}^{q-1} \mu_{bi}^{(l_k)} \right] \prod_{c \in N(i) \setminus \{a, b\}} \mu_{ci}^\neq \end{aligned} \quad (15)$$

$$\mu_{ia}^{(l_k)} = \prod_{c \in N(i) \setminus \{a\}} [\mu_{ci}^\neq + \mu_{ci}^{(g_{ia}^{-1} l_k)}] - \prod_{c \in N(i) \setminus \{a\}} \mu_{ci}^\neq \quad (16)$$

Finally, the marginals with respect to each z_i are found using the following equations:

$$P[z_i = *] \triangleq p_i^* = w_i \prod_{a \in N(i)} \mu_{ai}^* \quad (17)$$

$$\begin{aligned} P[z_i = l_k] \triangleq p_i^{(l_k)} &= \prod_{a \in N(i)} [\mu_{ai}^\neq + \mu_{ai}^{(l_k)}] - \prod_{a \in N(i)} \mu_{ai}^\neq - \\ &\quad - \sum_{b \in N(i)} \mu_{bi}^{(l_k)} \prod_{a \in N(i) \setminus \{b\}} \mu_{ai}^\neq, \end{aligned} \quad (18)$$

where $k = 0, 1, \dots, q-1$.

As a remark, the derivation of these message-passing rules is quite straightforward, despite of combinatorial nature. Due

³ $[\mu_{ja}^{(l_k)}]$ denotes the column vector whose q components are obtained for $k = 0, 1, \dots, q-1$; also, components of product of vectors are assumed to be products of the corresponding components. DFT_q and IDFT_q denote q -point Fourier transformation and its inverse, respectively. As a remark, we were able to derive the given formula for $[\lambda_{ai}^{(l_k)}]$ only for prime values of q .

to the limited space, the actual derivations were not included; however, one can notice how these message-updating rules appear as an extension of the ones relative to the $GF(2)$ field and derived in [5].

In practice, the quantization algorithm we designed follows the following steps:

- 1) All check-to-variable messages are initialized by assuming that there are no incoming variable-to-check messages; in particular, this implies $\mu_{ai}^* = w_s$, $\mu_{ai}^\neq = 0$, and $\mu_{ai}^{(l_k)} = \psi_a(g_{ia}l_k)$;
- 2) free variables that are either disconnected from any check or that connect exclusively to degree one checks are *removed* from the factor-graph;
- 3) message-passing rules for computing variable-to-check messages and recomputing check-to-variable messages are iteratively applied until convergence is achieved (or a maximum number of iterations is reached);
- 4) free variables that are sufficiently *biased* are *removed* from the factor-graph;
- 5) if there are still variables to be *removed*, go back to step 1; otherwise exit the algorithm.

Each time a free variable z_i is *removed*, its value is set as the l_k in $GF(q)$ that maximizes $p_i^{(l_k)}$; edges connecting that variable to checks are removed as well, and the compatibility functions ψ_a , for each $a \in N(i)$, are updated in order to take into account for this removal. Of course, unconnected checks are removed too from the factor-graph.

In addition, in order to quantize the *bias* of a probability distribution over $GF(q)^*$, we used the *unbiased* standard deviation of the vector $[p_i^{(l_k)}]$ multiplied by \sqrt{q} . In particular, if $\sum_{k=0}^{q-1} p_i^{(l_k)} = S$, i.e. $p_i^* = 1 - S$, it turns out that the bias B_i equals

$$B_i = S \sqrt{\frac{q \sum_{k=0}^{q-1} \left(\frac{p_i^{(l_k)}}{S} \right)^2 - 1}{q - 1}}, \quad (19)$$

which is always in the range $0 \leq B_i \leq 1$. As a remark, note that the maximum is achieved iff $S = 1$ (i.e. $p_i^* = 0$) and exactly one element of $[p_i^{(l_k)}]$ equals one. This formulation extends the bias concept introduced in [5].

IV. EXPERIMENTAL RESULTS

In the experiments we defined the quantizer $Q(\cdot)$ on the base of a parameter $0 < Q_m < q$, that specifies the minimum value in $GF(q)$ to be quantized as the non-zero value of $GF(2)$. In practice, $Q(l_k) = l_0$ for all $k < Q_m$, while $Q(l_k) = l_1$ otherwise. Hence, the employed GQ-LDGM codes have elements assuming the value l_1 with probability $r = \frac{q - Q_m}{q}$. We tested 4 different combinations of q and Q_m , as shown in Table I.

In these preliminary experiments, we derived the GQ-LDGM codes from randomly generated $GF(q)$ -LDGM codes with regular degree distributions. In particular, checks and free variables have degree d_c and d_v respectively, so that the resulting (source coding) rate, expressed in bit/sample units,

TABLE I
DEGREES, RATES AND SOURCE PROBABILITY USED IN THE VARIOUS EXPERIMENT SETS.

Experiment set	1	2	3	4
q	5	3	5	2
Q_m	4	2	3	1
r	0.200	0.333	0.400	0.500
d_c	2	2	2	2
d_v	9	6	9	4
R [bit/sample]	0.516	0.528	0.516	0.500
p_s	0.230	0.365	0.420	0.500

equals $R = \frac{d_c}{d_v} \log_2(q)$. We fixed the check degree $d_c = 2$ and derived, for each employed value of q , a suitable value of d_v for obtaining a rate of approximately 0.5 bit/sample, as shown in Table I.⁴

In order to use GQ-LDGM codeword statistics that match the optimal distribution, i.e. the one achieving the theoretical rate-distortion bound at the employed rates, in each experiment we generated a random i.i.d. source in $GF(2)$ with a probability of symbol l_1 equal to the values of p_s shown in Table I. These values are such that, approximately,

$$\frac{p_s - D(R; p_s)}{1 - 2D(R; p_s)} = r, \quad (20)$$

where $D(R; p_s)$ is the theoretical (Hamming) distortion at the employed rates R , according to the Bernoulli distortion-rate function [1]. In particular, D satisfies $H(D) = H(p_s) - R$, where $H(\cdot)$ is the entropy of the binary distribution with a given symbol probability.

In all experiments we assumed that the message-passing procedure relative to the various decimation steps converged when the maximum absolute difference between the (normalized) marginals $[p_i^*; p_i^{(l_k)}]$ computed in the current iteration and the ones computed in the previous one is less than $MP_{th} = 0.05$, for all free-variables. Also, we fixed the maximum number of iterations at $MP_{max} = 100$, which was typically reached in the first couple of decimation steps.

Similarly to what done in [5], after each message-passing run we removed at least $r_m = 1\%$ of the variables (starting from the more biased ones), as well as all the variables with a bias greater or equal than $B_m = 0.7$, until a maximum of $r_m = 10\%$ of the variables was removed. We also borrowed from [5] the choices of $w_i = \exp(0.05)$ and $w_s = \exp(0.10)$. Instead, in order to suitably choose β accordingly to the employed rate (and source statistics), we propose to choose it in a way such that the compatibility function $\psi_a(x_a)$ corresponds to the likelihood function of the *test channel* [1] related to the source coding problem at hand⁵; in practice, we

⁴As noted in [5] and [8], in the case of irregular degree distributions it is important to have an high ratio of checks of degree 2. This is the reason why we chose $d_c = 2$ in our experiments with regular distributions. Note that this choice, in case of $q = 2$, introduces linear dependency between the rows of the LDGM generating matrix. In practice, the actual rate could be (very slightly) reduced as there always exist 2 m -tuples of free-variables leading to the same LDGM codeword.

⁵This is reasonable as we may assume that the source (with probability p_s) is in fact a corrupted version of a codeword (with probability r), obtained after the latter went through the test channel (with probability of error $D(R; p_s)$).

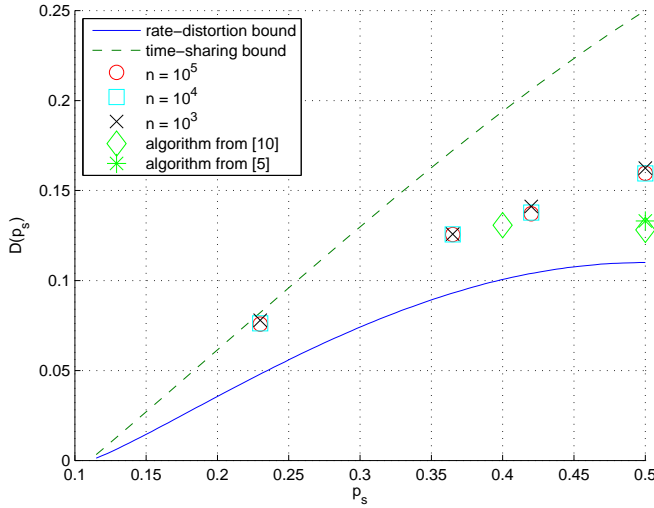


Fig. 2. Empirical performance at $R \cong 0.5$ bit/sample compared with the rate-distortion and time-sharing bounds and with some data from [10].

chose

$$\beta = \frac{1}{2} \log \frac{1 - D(R; p_s)}{D(R; p_s)}, \quad (21)$$

which equals approximately 1.53, 1.19, 1.10, and 1.05, respectively for each experiment set.

The results, reported in Fig. 2, are relative to compression of random realizations of length $n = 10^5$, 10^4 and 10^3 ; they have been averaged over 4 GQ-LDGM codes with 100 compressions/code, 10 codes with 500 compressions/code, and 10 codes with 1000 compressions/code, respectively for the three lengths. From the plot, where the rate-distortion bound $H(D) = H(p_s) - R$ and the time-sharing bound (between the zero-rate and the zero-distortion points)

$$D = p_s \frac{H(p_s) - R}{H(p_s)} \quad (22)$$

are given too, we concluded that the proposed algorithm performed quite close to the rate-distortion bound, and always better than the time-sharing bound. Surprisingly, the performance turned out to be essentially independent from the codeword length.

The algorithm was implemented in C and was run on an Intel Pentium 4 CPU operating at 3 GHz, with 1.5 GByte of RAM. In terms of execution speed it turned out to be very competitive with respect to the timings given in [5]. In fact, in the experiments of set 1, 80.53, 6.96 and 0.52 seconds were necessary on average for performing each source compression, respectively for realizations of length $n = 10^5$, 10^4 and 10^3 . In the experiments of set 2-4 the timings are 53.27, 3.98, and 0.31; 78.70, 6.93, and 0.54; and 46.74, 2.28, and 0.18, respectively.

V. CONCLUSION

In this paper we presented a novel message-passing/decimation algorithm for binary source quantization. The algorithm is based on the utilization of $GF(q)$ -quantized

LDGM codes, that permit to generate codewords with a non-uniform distribution. Eventually, this allows to directly compress sources with non-uniform distribution too. Among the applications, the proposed codes could be used in conjunction with good channel codes in problems of source coding with side information at the decoder.

The compression performances, obtained with regular degree distributions, show that the proposed algorithms perform reasonably close to the rate-distortion bound and always better than the time-sharing bound. It is also reasonable to assume that better performances could be achieved by suitably designing irregular degree-distribution codes, by augmenting the field cardinality q , or by employing damping strategies similar to the ones in [12] for solving convergence problems, which are possible directions for future research on this subject.

REFERENCES

- [1] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2006.
- [2] D. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.
- [3] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.
- [4] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.
- [5] M. J. Wainwright, E. Maneva, and E. Martinian, "Lossy source compression using low-density generator matrix codes: Analysis and algorithms," *IEEE Trans. Inf. Theory*, vol. 56, no. 3, pp. 1351–1368, 2010.
- [6] M. J. Wainwright and E. Martinian, "Low-density graph codes that are optimal for binning and coding with side information," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 1061–1079, Mar. 2009.
- [7] E. Martinian and J. S. Yedidia, "Iterative quantization using codes on graphs," in *Proc. of 41st Annual Allerton Conf. on Commun., Control and Comput.*, Oct. 2003, pp. 1317–1326.
- [8] P. A. Regalia, "A modified belief propagation algorithm for code word quantization," *IEEE Trans. Commun.*, vol. 57, no. 12, pp. 3513–3517, Dec. 2009.
- [9] M. Mézard, G. Parisi, and R. Zecchina, "Analytic and algorithmic solution of random satisfiability problems," *Science*, vol. 297, pp. 812–815, Aug. 2002.
- [10] A. Gupta and S. Verdú, "Nonlinear sparse-graph codes for lossy compression," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 1961–1975, May 2009.
- [11] E. Maneva, E. Mossel, and M. J. Wainwright, "A new look at survey propagation and its generalizations," *Journal of the ACM*, vol. 54, no. 4, Jul. 2007.
- [12] T. Filler and J. Fridrich, "Binary quantization using belief propagation with decimation over factor graphs of LDGM codes," in *Proc. of 45th Annual Allerton Conf. on Commun., Control and Comput.*, Sep. 2007, pp. 495–501.
- [13] A. Bennatan and D. Burshtein, "On the application of LDPC codes to arbitrary discrete-memoryless channels," *IEEE Trans. Inf. Theory*, vol. 50, no. 3, pp. 417–438, Mar. 2004.
- [14] A. Bennatan, D. Burshtein, G. Caire, and S. Shamai, "Superposition coding for side-information channels," *IEEE Trans. Inf. Theory*, vol. 52, no. 5, pp. 1872–1889, May 2006.
- [15] J. Chen, D. k. He, and A. Jagmohan, "Achieving the rate-distortion bound with linear codes," in *Proc. of IEEE Inf. Theory Workshop*, 2-6 Sep. 2007, pp. 662–667.
- [16] Z. Sun, M. Shao, J. Chen, K. M. Wong, and X. Wu, "Achieving the rate-distortion bound with low-density generator matrix codes," *IEEE Trans. Commun.*, vol. 58, no. 6, pp. 1643–1653, Jun. 2010.
- [17] Y. Zhang, S. Dumitrescu, J. Chen, and Z. Sun, "LDGM-based codes for successive refinement," in *Proc. of 47th Annual Allerton Conf. on Commun., Control and Comput.*, 30 Sep.-2 Oct. 2009, pp. 1518–1524.
- [18] —, "Multiple description coding based on LDGM codes," in *Proc. of IEEE Inf. Theory Workshop*, 11-16 Oct. 2009, pp. 85–89.